Project no. 027087

TENCompetence

Building the European Network for Lifelong Competence Development

Project acronym: Integrated Project TENCompetence

Thematic Priority: 2.4.10

D6.2 Compilation of internal deliverable outcomes ID6.4-ID6.5 (Learning activities and Units of Learning (main body of the report))

Due date of deliverable: 31-05-2008
Actual submission date: 12-01-2009

Start date of project: 01-12-2005
Duration: 4 years

University of Bolton

Version 1.0

<table>
<thead>
<tr>
<th>Dissemination Level</th>
<th>PU</th>
<th>PP</th>
<th>RE</th>
<th>CO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public</td>
<td>Restricted to other programme participants (including the Commission Services)</td>
<td>Restricted to a group specified by the consortium (including the Commission Services)</td>
<td>Confidential, only for members of the consortium (including the Commission Services)</td>
<td>X</td>
</tr>
</tbody>
</table>

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)
Project Deliverable Report

D6.2 – Compilation of internal deliverable outcomes ID6.4-ID6.5 (Learning activities and Units of Learning (main body of the report))

Work package
WP6 Learning Activities & Units of Learning

Task

Date of delivery
Contractual: 31-05-2008
Actual: 12-01-2009

Code name
D6.2
Version: 1.0
Draft ☐ Final ☒

Type of milestone
Report

Security (distribution level)
Public

Contributors
Atanas Georgiev, David Griffiths, Davinia Hernandez, Javier Melero, Krassen, Stefanov, Mar Perez, Michele Dicerto, Paul Sharples, Phillip Beauvoir, Scott Wilson, Tim Sodhi, Yongwu Miao

Authors (Partner)
University of Bolton, OUNL, SU, FBM-UPF, ILABS

Contact Person
David Griffiths (University of Bolton)

WP/Task responsible
David Griffiths (University of Bolton)

EC Project Officer
Mr. Martin Majék

Abstract (for dissemination)
The report describes the work of the TENCompetence project to provide a standards based open source infrastructure to support the implementation and running of learning and assessment activities. The TENCompetence Learning Design Toolkit is presented, and its components described. These include the ReCourse authoring tool with its associated plug-ins, and a set of runtime applications, including extended and enhanced versions of CopperCore, APIS, SleD. Of particular significance is the development of an entirely new widget server to resolve the longstanding problem of services in IMS LD.

The Assessment Process Specification is described. This is a meta model of assessment processes which can be expressed as a combination of IMS LD and QTI. This has been achieved by modifying the OUNL/CITO assessment model from the perspective of process support.

Evaluation reports are submitted in an annex to this deliverable

Keywords List
IMS Learning Design Authoring, assessment specification, run-time IMS LD services, Units of Learning, ReCourse, Widget server

TENCompetence Project Coordination at: Open University of the Netherlands
Valkenburgerweg 177, 6419 AT Heerlen, The Netherlands
Tel: +31 45 5762624 – Fax: +31 45 5762800
TABLE OF CONTENTS

1 EXECUTIVE SUMMARY .............................................................................................. 2

2 OVERVIEW .................................................................................................................. 4
   2.1 THE TENCompetence LEARNING DESIGN TOOLKIT ........................................... 4
   2.2 TENCompetence ASSESSMENT SPECIFICATION VERSION 2 ......................... 6
   2.3 ASSESSMENT OF PROGRESS .............................................................................. 6

3 THE SIGNIFICANCE OF THE WORK CARRIED OUT .............................................. 9
   3.1 The significance of the WP Core System in the TENCompetence project .......... 9
   3.2 The wider significance of the WP6 core system .................................................. 10

4 COMPONENTS OF THE CORE SYSTEM .................................................................... 12

5 AUTHORING: THE RECOUSE LEARNING DESIGN EDITOR .................................... 14
   5.1 INTRODUCTION .................................................................................................. 14
   5.2 DESCRIPTION AND RATIONALE OF CHANGES TO AUTHORING INTERFACE SINCE D6.1 ................................................................. 16
   5.3 LINK WITH OPENDOCUMENT.NET REPOSITORY ............................................... 18
   5.4 AUTHORING OF SERVICES (LINK WITH WIDGET SERVER) ............................... 21
   5.5 PROVISIONING .................................................................................................. 24
   5.6 THE QUESTIONNAIRE AND TEST PLUG-IN ..................................................... 25

6 RUNTIME: WIDGET SERVICES FOR LEARNING DESIGN ...................................... 28
   6.1 INTRODUCTION .................................................................................................. 28
   6.2 THE TOOLS ......................................................................................................... 28
   6.3 LINKING AN IMS-LD SERVICE TO A CONCRETE WIDGET SERVICE ................ 29
   6.4 WIDGET CONTEXTS AND WIDGET DEFAULTS ................................................. 30
   6.5 IMPLEMENTATION OF THE WIDGET SERVER ................................................ 30
   6.6 UPDATES TO OTHER PIECES OF SOFTWARE .................................................... 31
   6.7 THE INTERACTION CYCLE OF LD RUNTIME AND WIDGETS ......................... 32
   6.8 EXTENDING THE WIDGET SERVER .................................................................. 33
   6.9 WIDGET ADVERTISING ...................................................................................... 34
   6.10 SERVICE ROLES .............................................................................................. 34
   6.11 LINK FOR SOFTWARE ...................................................................................... 35
   6.12 OTHER RUNTIME DEVELOPMENT .................................................................. 35

7 ASSESSMENT ............................................................................................................... 39
   7.1 INTRODUCTION ................................................................................................. 39
   7.2 THE PROCESS ORIENTED APPROACH TO MODIFYING THE OUNL/CITO MODEL .......................................................... 39
   7.3 THE ASSESSMENT PROCESS SPECIFICATION (APS) ....................................... 40

Note that an annex containing evaluation reports is submitted as a separate document.
1 Executive summary

In TENCompetence WP6 has the task of adapting and developing tools open source and standards based tools for authoring and running learning activities to support the requirements of lifelong competence development. As described in the Detailed Implementation Plan 2, this report delivers the second release of the TENCompetence prototype tools developed by WP6 to meet this goal. The design and implementation of this toolkit was guided by D2.2 Updated use case models and underlying vision documents and pedagogical model definitions. This defined the core functionality to be provided by all TENCompetence work packages in order to support the project vision and pilots, and which would provide a solid platform for further development in the remainder of the project. Within this framework WP6 was allocated two usage profiles to address create course and run course, with a focus on integration of functionality into a usable system.

The proof of concept tools in this release are together referred to as the TENCompetence Learning Design Toolkit. It may divided into two main sections, authoring and runtime, but these are closely integrated. The target users are expected to have some background in technology enhanced learning (for example learning designers and teachers who have an interest in pedagogic modelling) but are not required to have advanced technical expertise in order to use the software.

The toolkit enables users to edit and publish and run units of learning in a much easier and more flexible way than has previously been possible. The WP6 prototype tools delivered in D6.1 consisted of an IMS-LD editor, ReCourse, a set of assessment tools, and a version of the SLeD LD player which integrated SCORM and IMS-LD. These components were, however, either not integrated, or required some expert input to achieve an integration. The present Learning Design Toolkit, however, marks an advance in providing an integrated environment for IMS-LD authoring, set up and delivery of learning activities which is significantly better integrated and easier to use than any existing IMS-LD compliant system. It also provides a solution developed by the project to the longstanding problem of providing a flexible set of runtime services for IMS-LD, and published as the Wookie Widget Server. In addition the interface for authoring Units of Learning has undergone a major revision in response to user feedback.

This toolkit has been evaluated with users in Liverpool, Sofia and Barcelona to ensure that the system performs well, and in order to identify outstanding usability problems, and a report is submitted as an annex to this deliverable. This showed that teachers can use the system to define Units of Learning provided that they are given sufficient training (estimated at 4 to 8 hours). Two principal problems were experienced by users: a) management of resources was complex and confusing, b) some of the concepts of IMS Learning Design were unfamiliar. The management of resources has been transformed in the latest release and initial feedback suggests that it is far easier to use. None of the concepts of IMS Learning Design proved difficult to explain to users.

---

1 D2.2 Updated use case models and underlying vision documents and pedagogical model definitions is available at http://hdl.handle.net/1820/1152
users, and while improvements have been made in clarifying the interface in this respect, the preparation of improved support materials and templates is the principal need. The toolkit is a very extensive system, and investigation of the way in which the system is used and understood will require longer term evaluation. It is planned to carry this out in a business demonstrator which applies the toolkit to the development of business simulations.

D6.1 described how work with the first release of the proof of concept authoring tools showed that the first version of the TENCompetence Assessment Specification could not be satisfactorily exported to a combination of IMS-LD and QTI as required by project strategy\(^2\). In order to repair this problem a new version of the assessment specification was developed, called the TENCompetence Assessment Process Specification in order to distinguish it from its predecessor. This enables the user to define Units of Assessment which provide more sophisticated competence assessments than those supported by IMS-QTI. The Assessment Process Specification has been tested by representing a complex assessment process defined with the specification as a combination of IMS-LD and IMS-QTI.

Proof of concept authoring and runtime support for the new assessment specification has been put in place in this period, and is integrated with the Learning Design Toolkit. In this way authoring of Units of Assessment is supported by the same infrastructure as that used for authoring Units of Learning. This has been tested by implementing and running a complex example of a Unit of Assessment, peer evaluation. Units of Assessment can be run using the adapted SLeD and APIS servers available in the toolkit. An IMS-QTI component has been developed and integrated into the Toolkit which provides support for the elements which are needed to express the Assessment Process Specification. The need to develop this component was not planned for in the project plan, because it was hoped that one would become available. The need to develop one has been responsible for the delay in the submission of this deliverable, but has not prevented work continuing on other aspects of this work package.

Authoring of Units of Assessment in this release requires some expert skills because the authoring system does not provide an interface for authoring the IMS LD level B properties required for representing some aspects of the Assessment Process Specification. These currently have to be inserted using the Reload LD Editor, or a similar application. This is, however, an interface issue (since as all the required data structures are handled by the system) and will be addressed in the following release.

The Learning Design Toolkit delivered here is a significant achievement, providing much needed functionality for the authoring and delivery of standards based learning activities. It also provides a solid software platform which will be built on in the next release of the system.

The Toolkit is available for download as two easy to install packages together with documentation at
http://www.tencompetence.org/ldauthor/ and
http://www.tencompetence.org/ldruntime/

\(^2\) The D6.1 summary report is available at http://hdl.handle.net/1820/1148, and Annex 2: Assessment is available at http://hdl.handle.net/1820/1150
2 Overview

This deliverable marks the publication of two significant TENCompetence WP6 products:

a) **The TENCompetence Learning Design Toolkit.** This is the core system developed by WP6, and it supports users in working with IMS-LD Units of Learning. It provides much improved integration of authoring, storing, publishing, allocation of learners, and running of learning activities.

b) The **TENCompetence Assessment Process Specification** has been developed, and tested. Runtime support and expert authoring tooling for the specification is in place.

In this section we provide an overview of the main characteristics of these.

2.1 The TENCompetence Learning design Toolkit

At the vision group meeting in Dagstuhl and the following project meeting in Maastricht priority in all project work was given to the core system, scheduled for delivery in May 2008. This strategy was then formalised in D2.2 Updated use case models and underlying vision documents and pedagogical model definitions

In accordance with this strategy workpackage 6 has therefore focused its work on the two usage profiles assigned to it in D2.2: create course and run course. These usage profiles are described in detail in D2.2, and so are not reproduced here.

The WP6 part of the TENCompetence core system is published in this deliverable as the TENCompetence Learning Design Toolkit, providing integrated functionality for both profiles. It addresses both the create course and run course profiles in two integrated sets of software, for authoring and running Units of Learning (UOLs) and Units of Assessment defined using IMS Learning Design (IMS LD) and IMS QTI (Question and Test Interoperability). The target users are both technical experts and those teachers who have an interest in pedagogic modelling and learning design. Advanced technical expertise is not required to use the software.

The authoring component is built around ReCourse, a Java application developed by the project as a successor to the Reload LD Editor. Version 1.0 of ReCourse was delivered at the end of 2007. Since that release consultation with the user group (in particular the IMS-LD authors at Associate Partner Liverpool Hope University) has led to major changes to the interface, with a new focus on activities as the core of the authoring process. Following evaluation results the handling of resources had been transformed, with default files created for users, drag and drop of external resources, automatic management of resource names so that they can be related to source files. The provision of the QTI 2.1 compliant questionnaire and test authoring tool rectifies the problem of separation of assessment and learning activities which has its origins in the adaptation of OUNL Educational Modelling Language to create IMS Learning

---

3 D2.2 Updated use case models and underlying vision documents and pedagogical model definitions is available at http://hdl.handle.net/1820/1152
Design. Support for Level B and C is complete at the file level and in the runtime, and an inspector and editor has been implemented for Level B properties.

A significant advance has been that the application now strongly leverages the ReCourse plug-in architecture, which is inherited directly from the Eclipse framework. In this way the application becomes a framework for the development of an extensible authoring environment. Existing plug-ins comprise:

- **Rich Text Editor**, used in authoring resources.
- **OpenDocument.net repository component**. This enables authors to browse Units of Learning and to upload their work. The repository can parse the Units of Learning which it stores, and return information about their contents (e.g. learning objectives). This means that authors can get basic information about the Unit of Learning without downloading it.
- **Fedora Connector**, which enables authors to upload their Units of Learning to the TENCompetence repository from within ReCourse.
- **Widget Server discovery**. This links with the TENCompetence Wookie Widget server, and provides information on the runtime services available to authors.
- **CopperCore Manager**, providing the first easy to use means of publishing Units of Learning, creating a run, and populating it with learners. Previously this task had to be carried out with a command line interface to the server.
- **Questionnaire and test component**, which generates IMS QTI 2.1.

A major step forward has been taken in the **runtime component** with the development and forthcoming release of the Wookie widget server, which integrates with the existing CopperCore and SLeD servers. This provides a solution to the long standing challenge of providing flexible runtime services for Units of Learning. In the present release widgets for chat, forum, voting, and Google maps are included, but more widgets can expected to be adapted and authored soon. The use of a widget layer to resolve this problem in IMS LD is an innovative solution, as is the creation of a server which can handle collaborative widgets.

The APIS QTI engine has been extended and enhanced, to ensure smooth integration with the questionnaire and test authoring component.

The core system was demonstrated at a JISC Design for Learning meeting on 20th May, where it generated a lot of interest among the UK community. Evaluation of the system is currently underway in Sofia and Barcelona.

The core system has been evaluated in trials carried out with learning designers and teachers in Liverpool Sofia and Barcelona, and the results are provided as an annex to this deliverable (separate document). The evaluation focused in particular on authoring, and concluded that teachers were able to work with the Toolkit provided that a modest amount of training was provided of between 4 and 8 hours. A number of usability issues were identified, including a major problem with authoring and managing resources which led to major changes in the application.

Further information and downloads are available at the following links
- ReCourse LD Editor [http://www.tencompetence.org/ldauthor/](http://www.tencompetence.org/ldauthor/)
2.2 TENCompetence Assessment Specification Version 2.

The assessment strategy adopted by the project has been to develop an assessment specification which can be used to define sophisticated assessment processes, and to author and deliver the resulting units of assessment using a combination of IMS LD and QTI. As described in D6.1, evaluation of the first version of the Assessment Specification showed that it had serious shortcomings which meant that Units of Assessment developed with it could not be expressed as a flows of activities using IMS LD. In the phase of work reported here the Assessment Specification has undergone radical modification, and Version 2 has been released as the TENCompetence Assessment Process Specification (APS).

The Assessment Process Specification is expressed as a combination of IMS LD and IMS QTI 2.1, has been successfully tested using a complex example of a Unit of Assessment (peer assessment). Consequently it was clear that it would be wasteful to create a specialised authoring infrastructure for Units of Assessment, separate from that for Units of Learning. A search was carried out for an open source QTI 2.1 editor which met the software requirements of the project. This included meetings with the JISC AQuRate project, who are also developing for QTI 2.1. However, no appropriate software was identified, and as a result development resources were diverted to the development of QTI a component for ReCourse. This is now in place, and the combined system constitutes the proof of concept authoring environment for Assessment Specification v.2. At present this approach has limitations, because the authoring interface for IMS LD levels B & C is not yet in place in ReCourse, and this is required for expression of some aspects of the Assessment Specification. As a result authoring Units of Assessment requires some expert level authoring skills. However, this is primarily a matter of interface design, as both the authoring and runtime components handle all the required files satisfactorily, and this limitation will be resolved in the next stage of project work.

2.3 Assessment of progress

The applications scheduled for this release of the WP6 tools have three components: IMS LD Authoring, IMS LD Runtime, Assessment Authoring. The present deliverable represents substantial progress in all three aspects, and integrates all three into a coherent architecture. Indeed, the TENCompetence Learning Design Toolkit is the first system to provide an integrated set of tools for non technical experts which

- covers the whole process of authoring, storing, browsing, setting up, provisioning and running Units of Learning compliant with IMS LD
- provides flexible and extensible set of services
- makes substantial advances in usability

In line with the project policy of focusing on provision of a core infrastructure, the decision was taken to integrate the infrastructure for assessment and learning activities authoring into a single framework. This was made possible by the revision of the

See the AQuRate website at http://aqurate.kingston.ac.uk/
TENCompetence Assessment Specification so that it could be expressed by a combination of IMS LD and QTI.

As a result the authoring environment provides tools for authoring both IMS LD and the profile of IMS QTI which was required by the Assessment Specification V.2. This approach has the advantage of ensuring integration of the components, and enables the team to focus on the development of a single set of high quality tools. Similarly the runtime environment for learning activities and assessment has been integrated. One drawback of this integrated approach has been that it has increased the dependencies between work on the different components. As a result the authoring environment for assessment activities has been constrained in the short term, as the authoring interface for IMS-LD is only now becoming available. Consequently expert authoring skills are at present required to develop Units of Assessment which require complex or conditional workflows.

The development of the Learning Design Toolkit is an essential step for the project, which has committed to IMS LD as the enabling technology for delivery of competence development activities. The system fulfils all the requirements established for the core system following the Maastricht meeting in October 2007. The Toolkit provides a solid base of functionality, which can be leveraged in the development of more focused tools for specific pedagogic and assessment approaches in the next phase of the project.

The priorities identified for the next phase of implementation work are

- Extended support for authoring IMS LD levels B & C. The runtime component executes all three levels of the specification, and the authoring component handles levels A, B & C at the data level. However authoring support is currently limited to an inspector and editor for Level B properties. As it stands is a viable solution for expert users who want to author Level B, but more support will be provided for authors, including automated support for the most common use cases (e.g. branching structures according to the results of tests) and a built in XHTML editor for creating IMS LD Content.

- Provide extended support for templates, and a set of templates which can be used by authors. This will be for two types of authors. Firstly, teachers and learning designers who want to quickly and easily create Units of Learning, and secondly for more expert users who will be assisted in the creation of more complex UOLs by providing a structure within which the properties and conditions which control the learning or assessment flow can be defined.

- Further develop the QTI editor in two aspects
  - add three interaction types (e.g., slider, order, and match) which the new APIS can support
  - A list view of items which can be dragged onto tests
  - Rich Text Editing of questions
  - Item banking to support the creation of tests

- Develop user-friendly response-processing templates and support the integration of QTI outcome variables with LD properties.
• Develop a TENCompetence component which can take the place of SLeD. SLeD is the IMS LD player which users interact with in following IMS LD learning activities. There are two principal problems with this
  o the interface is a rigid and old fashioned division of the screen into panes
  o the code has been developed in a succession of projects by various developers using different approaches. It is therefore not a solid base to build on.

It is intended to create a substitute for SLeD, building on the key concepts which have been identified in working with SLeD over the last two years, and new approaches proposed by the WP6 team.
3 The significance of the work carried out

3.1 The significance of the WP Core System in the TENCompetence project

In order to maximise the coherence of project development work as a whole, the WP6 core system addresses the use cases are set out in D2.2 Updated high level Use Cases and pedagogical model definitions.

The relevant usage profiles are

a) “Follow course” (D2.2 p.38)
b) “Create Course” (D2.2 p.44)

Analysis of the functionality required to support these two usage profiles led to the definition of the WP6 core functionality delivered in this report. Particular stress was given to support of learning designers and teachers in the whole process from browsing and storing UOLs, editing, and publishing and running learning activities.

The scope of the system is limited to learning and assessment activities defined using IMS LD and QTI. Simpler Competence Development Plans (a set of objectives, resources and activity descriptions which address a competence) are also supported by the Personal Competence Manager (PCM) system, and can be edited with the PCM Rich Client, and by the web based PDP tool. These simple competence development activities do not have support for cohorts, collaboration, teacher engagement or complex learning flows.

TENCompetence requires an effective means of authoring and delivering pedagogically flexible learning activities, using open source and standards compliant infrastructure. Some valuable applications were available in this respect at the start of the project, notably CopperCore, SLeD and Reload, and APIS, and these have been leveraged by the project. However the functionality available was well short of that which was required, and the work submitted in this deliverable is a significant step towards updating and extending this tool set to create an effective open source infrastructure for creating and running Units of Learning and Units of Assessment. The result is an integrated system with sufficient power and flexibility that it can be used in project pilots as the principal means of delivering competence development activities.

---

5 D2.2 Updated use case models and underlying vision documents and pedagogical model definitions is available at http://hdl.handle.net/1820/1152
3.2 The wider significance of the WP6 core system

In meeting the requirements of the project, the WP6 core system has had to address a number of long standing problems with IMS Learning Design

1. the lack of runtime services
2. difficulties in publishing and populating learning activities
3. separation of assessment from learning activities

3.2.1 Providing flexible runtime services in IMS Learning Design

IMS Learning Design does not describe a full set of tools that could be used within a learning context, such as a wiki or forum. Rather, it specifies a small subset of generic service types that could be used within a learning context. This is reasonable, as the services used are developing rapidly, and future requirements are inevitably unknown. Additionally, the specification leaves open how the design is to be realised in runtime software. Consequently the question “what is a conference?” arises. Is it voice chat, instant messaging, video conference or something else?

This problems which this approach creates were recognised when the specification was developed, as recognised by Olivier, one of the authors of the specification (Olivier & Tattersall, 2005)

Clearly many more services could be added to the LD specification, and it is desirable that they should be, from chat, instant messaging and white boards, through virtual classrooms and more sophisticated collaborative services, such as virtual design environments, to sophisticated simulation and multi-user game-playing systems.

The key issue that needs to be addressed is how to add services in such a way that key learning designs that use them still retain a reasonable degree of portability across different LD-compliant platforms. If all the above services were included, could any system be expected to be compliant? Or should the specification stick to the lowest common denominator for services...?

This key issue has resisted solution since IMS-LD was published in 2003, leaving users to choose between systems which provided rich runtime services but were not fully interoperable (for example LAMS) or fully compliant IMS-LD systems which provided only a limited set of services (such as SLeD). The runtime environment delivered here is the first system which maintains the structure of the IMS LD specification but also provides flexible services, and as such is a significant achievement of the project.

At another level the integration of CopperCore into Giunti’s flagship eLex platform has also demonstrated how the open source services provided by TENCompetence can be leveraged by organisations which depend on the use of proprietary enterprise systems.
3.2.2 difficulties in publishing and populating learning activities

Prior to this release of the WP6 core tools, in order to publish a Unit of Learning to CopperCore and make it available to learners it was necessary to validate and publish through the CopperCore web interface, and then use the Command Line Interface CopperCore to set up runs and populate them with learners. This proved hard for non-technical users to carry out, and meant that technical staff were required to carry out this basic functionality. The TENCompetence Learning Design Toolkit provides a much simpler publishing and provisioning process, fully integrated into the authoring environment. The system works with a remote server (for providing learning activities to learners) or in the local loop (so that authors can preview Units of Learning).

3.2.3 separation of assessment from learning activities

When Open University of the Netherlands Educational Modelling Language was adapted to create IMS LD, all the parts which related to questionnaires and tests were removed. This was in order to avoid duplication with the pre-existing IMS Question and Test Interoperability specification. Unfortunately this created in the infrastructure an artificial division between learning activities and assessment activities. The integration of a question and test authoring component in this release of the ReCourse Learning Design Editor is a substantial step towards resolving this problem. This solution, combined with the updated APIS QTI runtime system, provides the basis for tests integrated with units of learning which enable the learner to assess their knowledge or evaluate their progress. By using Level B, Learning flows can be defined which are conditional on the results of learners’ performance in the tests. In the next phase of development interfaces will be provided which make such structures easier to author.

The way in which the system resolves the three problems outlined above makes it of immediate relevance to a wider community beyond the confines of the project. Consequently (in addition to normal dissemination activities) it has already been presented to JISC in the UK, to the IMS Summit in September 2008, and at workshops in ICALT 2008 in July and ALT-C 2008 in September.
4 Components of the Core System

The WP6 core is an integrated system enabling teachers to create, publish and populate courses. The authoring components are Rich Client, and the end user runtime systems are web based.

The services used are

(a) CopperCore
(b) SLeD
(c) the TENCompetence Wookie Widget server v.1.
(d) the OpenDock repository
(e) the APIS QTI 2.1 runtime engine
(f) the TENCompetence Fedora repository

The runtime interface is a revised version of the SLeD player

The authoring components have been developed in Java, using the Eclipse Rich Client Platform, and are integrated in the ReCourse application developed by WP6.

The principal components of the core system are as follows:

1. Authoring
   1.1. Extended ReCourse IMS LD editor and plug-in framework, v. 1.5.2
   1.2. QTI authoring plug-in with, (1) multiple choice (2) multiple response (3) fill in the blank (4) Likert scale. Creates QTI 2.1
   1.3. Integrated authoring of services, selected from list made available to the author by the Widget server.
2. Publishing
   2.1. Publishing component created as an Eclipse plug-in. Publishes to CopperCore, and provides an interface for setting up runs and users.
   2.2. Integration with Opendocument.net repository. Browsing by parsed manifest, upload, download.
   2.3. Integration with the TENCompetence Fedora repository
3. Runtime
   3.1. The Wookie Widget server, providing flexible runtime services for IMS LD, with administrator module.
   3.2. New version of APIS questionnaire and test runtime, updated to IMS QTI 2.1
   3.3. New version of SLeD Learning Design Player, with updated interface, and integrated APIS and Wookie server
   3.4. Integrated “one click” installer for CopperCore, SLeD, Wookie and APIS
The main body of this report is a description of this system, which is available as binary installers on the TENCompetence public site\textsuperscript{6}, with the source code published on SourceForge\textsuperscript{7}.

\textsuperscript{6} http://www.tencompetence.org/ldruntime/ and http://www.tencompetence.org/ldauthor/

\textsuperscript{7} http://tencompetence.cvs.sourceforge.net/viewvc/tencompetence/wp6/
5 Authoring: the ReCourse Learning Design Editor

5.1 Introduction

ReCourse is an authoring tool for the creation of IMS Learning Designs which was developed in the first phase of work on WP6, as described in D6.1. In the present deliverable functionality of ReCourse has been extended, but moreover the plug-in architecture established has been leveraged to create the environment where the author of learning activities can manage the whole workflow of searching, authoring, storing and setting up runtime learning activities.

We now review the main characteristics of ReCourse. The main intent of the application is to allow the creation IMS Learning Designs (UOLs) in an easy and intuitive manner. It is based on the Reload LD Editor with new design ideas and drawing on Bill Olivier’s PowerPoint document, “From EML to Learning Design & Planned OS Implementations focusing on RELOAD” and from contributions from Bolton University, Liverpool Hope University and the OUNL. The main theme is simplicity and a task-focused UI that reduces information overload. Whereas the intent of the original Reload LD tool was to expose all of the many LD elements in trees and tables and thereby expose the full extent of the IMS LD specification, the intent of the new tool is to expose only the necessary elements at a structural level broadening the focus and to provide a meaningful perspective on the designer’s intentions.

Input

ReCourse is able to open any IMS LD compliant packages, parse them and present them to the user for editing in the user interface (UI). Once opened they can be edited and re-saved or uploaded to a repository. It can open LDs from other users of the tool. The IMS XML is preserved at all times.

In addition to this, the ReCourse QTI plug-in can open files which conform to the subset of QTI 2.1 used by TENCompetence.

ReCourse can also acquire data by

a) querying the Wookie Widget server, in order to display a list of available services for use in a Unit of Learning

b) querying an OpenDocument repository, and displaying information about the UOLs and other information stored there

Output

ReCourse can open and save full UOLs at IMS LD levels A, B & C, while the authoring interface allows the creation of IMS Learning Designs at Level A, with support for experts at Level B. A full Unit of Learning conforms to the IMS LD spec and XML binding, with its imsmanifest.xml file and content packaged in an IMS Content Package (a “Unit of Learning. Additional information for the LD, such as UI information (colours, positions of graphical elements, etc) descriptions, notes and narratives are saved in the imsmanifest.xml file. This will only have meaning for the recipient if they re-edit the UoL in their copy of ReCourse.
Eclipse RCP

ReCourse has been created as an Eclipse-based Rich Client Platform (RCP) desktop application. This approach benefits from the following main advantages:

- Java based
- Native UI
- Plug-in architecture
- Extension points for add-ons
- Open Source

The use of the Java language as the basis for RCP means that the application can be built against a number of target platforms. ReCourse is currently built against the following:

- Windows 32-bit (XP, Vista)
- Mac OS X (Universal Binary)
- Linux 32-bit
- Linux 64-bit

Eclipse uses the Standard Widget Toolkit (SWT), a graphical widget toolkit for use with the Java platform. This means that the look and feel of the application remains true to the host’s platform user interface.

Plug-in architecture

The plug-in architecture for ReCourse is inherited directly from the Eclipse framework on which it is built. This architecture ensures that the core components of the application themselves are modular, and therefore more readily maintainable, and that additional features may be included without impacting on the core code.

Existing plug-ins comprise:

- Rich Text Editor
- OpenDocument.net repository search and upload
- Widget Server discovery
- Publish to and populate CopperCore Server
- Questionnaire and test editor

Graphical information

Since the XML document for the IMS Learning Design (imsmanifest.xml) does not capture additional information such as the designer’s notes, UI positioning data and so on ReCourse stores this information at a name spaced extension in the “metadata” tag of the manifest file. This is bound against a schema (XSD) file that can be optionally included with the Unit of Learning package.
5.2 Description and rationale of changes to authoring interface since D6.1

The first version of ReCourse relied heavily on the use of the Eclipse Graphical Editing Framework (GEF) to provide a user interface inspired by the UML design approach. This early version of ReCourse divided the design and editing areas into the following panels:

- Course – corresponding to the “Method” of Learning Design (Plays, Acts and Role Parts)
- Activities
- Environments
- Roles

Some example screenshots illustrate this approach:

Figure 1: The Original “Course” editor panel
However, using the editor in real situations and additional feedback from users showed that this approach was not entirely suitable for certain sub-domain models. For example, Roles are usually limited to one or two in number (learner and teacher) and this does not require a full diagram editor. Likewise, Activities and Activity Structures are more suited to be presented to the user in a more traditional hierarchical tree graph.

The main focus of a Learning Design at design time in the latest version of ReCourse is at the “Method” level with the user able to quickly add and organise Activities and Roles in a “Role Part” table. The designer can now simply add a new Activity or Activity Group directly as a Role Part, and the underlying components are added at the same time. The intent now is that the user should be able to access most elements from one editing panel. Indeed, the new “Course” panel offers the user the ability to create and edit Plays, Acts, Role Parts, Activities and Activity Groups, and Roles all from one pane. Furthermore there is readily available links to provide the overview, prerequisites, learning objectives, completion rules and the final packaging of the Unit of Learning all from one place.

ReCourse now includes an “Organiser” view that supports the organisation of various granularities of Learning Design artefacts. The largest granularity is the Unit of Learning itself where the designer can store, tag and organise these at a local file
system level. As each Unit of Learning is opened in ReCourse its components, consisting of Activities, Environments and Roles can be accessed in the Organiser View to be re-organised, edited, added and deleted.

![Figure 3: The revised “Course” editor panel.](image)

The development team believe that the GEF approach may become valuable at IMS LD levels B in the representation of branching workflows, for example in a visual properties and conditions editor.

### 5.3 Link with OpenDocument.net repository

ReCourse includes a plug-in that connects to a given OpenDocument.net repository providing the features of:

- Search and retrieve Units of Learning
- Upload Units of Learning

There are two different usage profiles for a repository in relation to Units of Learning:

a) Collaboration in the creation of UoLs. Somewhere to store, browse and use resources, components of UoLs, part finished or hacked UoLs, etc.

b) A place where the UoLs to be used in TENCompetence Competence Development Plans can be stored.
An OpenDocument.net repository can provide for the first usage profile, while the second is handled by the repository provided by WP5.

In ReCourse, the URL and ID of a given repository can be provided from the Preferences menu (Edit->Preferences) or from the toolbar button in the OpenDocument.net View. Search criteria are provided by the user in the Search panel:

![OpenDocument.net search panel](image)

**Figure 4: The OpenDocument.net search panel**
Double-clicking on an item in the Search table allows the user to view further details of the UoL and download it (perhaps for additional editing) if required:

![Creative Writing UoL Details](image)

**Figure 5: The OpenDocument.net search results view**
Conversely, the ReCourse user is able to upload a finished Unit of Learning to a given repository and provide some covering meta-data:

![Figure 6: Upload to OpenDocument.net repository](image)

5.4 Authoring of services (link with Widget server)

IMS Learning Design describes and supports four types of abstract services – send mail, conference, index search and monitor. Two of these, send mail and conference, will be instantiated at runtime with a concrete implementation. The CopperCore runtime system has been extended with a “Widget Server” that provides these services manifesting as desktop widgets. In order for the designer to be aware of what widget services are available for a given server, and to provide a convenient method of adding these services to an Environment in ReCourse, there is an additional plug-in that forms the link between the editor and the Widget runtime system.
This feature surfaces in ReCourse under the “Tools” menu. The user provides the URL of the Widget server in the dialog box that appears:

![Query Widget Server dialog box](image)

Figure 7: Widget Server query dialog box
If successful, the user is shown a message indicating what Widgets are available in the Editor. The current editing session will now support the addition of these widgets as Services to the Unit of Learning in the Environment's editor where they will appear in the palette:

![Palette with Widgets](image)

**Figure 8: Services shown in the authoring palette**

Technically, the underlying action is simple. The “parameters” attribute of a service is allocated a string such as “widget=chat”. This indicates to the server at runtime that the given service requires an instantiation of a particular widget. However, this does provide to the editor, a visual cue for what services are available in real terms, rather than the abstraction of a “conference”, for example.
5.5 Provisioning

Units of Learning can be uploaded and published to a CopperCore server from within the ReCourse editor. UoLs can be managed; “Runs” and Users can be created and deleted. This is achieved by opening the Publisher View from the "Window->Show View" menu or from the top toolbar button:

![Figure 9: The publisher View](image)

The UoL will be saved to a zip file first as part of the provided packaging wizard or may already exist as a zip file.

The toolbar of the Publisher View has

- A Combo box where the user can enter the URL of the CopperCore server if required, or a default may be used. The “Connect” button to connect to the server.
- The Upload button allows the user to upload the zipped UoL to the CopperCore server.
- UoLs that exist on the server appear in the first pane. These can be deleted. Additional UoLs are uploaded from the Upload button.
- Runs can be added and deleted from the Runs pane and the Add Run button on the toolbar.
- Roles appear in the Roles pane. These are the Roles present in the UoL. These cannot be deleted or changed as they are fixed in the UoL.
- Users can be added and removed in the Users pane. Note - these show the active users for the Run.
5.6 The Questionnaire and Test Plug-in

IMS QTI describes a data model for the representation of assessment item/test and their results. It defines a set of interaction types which can be used to define basic question types (e.g., multiple-choice, match, and filling_in_blank) and complicated question types through combination. Within TENCompetence these question types are not only used in classical assessment types covered by IMS QTI, but also in new forms of assessment established in the TENCompetence Assessment Specification v2, submitted in this deliverable. There is a need for an authoring tool which can meet both these aspects.

We investigated whether existing QTI v2 authoring tools (e.g. the AQuRate tool8) could be used for our purpose, but none were suitable. We believe that this was because:

- IMS QTI v2 is powerful and technically complicated, it is very difficult to develop an authoring tool for the full specification for ordinary practitioners (not for technical experts).
- IMS QTI v2 is a recent specification
- In order to meet the software specifications of TENCompetence we needed an open source Java tool implemented in Eclipse

As a result, it was necessary to develop an IMS QTI authoring tool to cover those question types required for the TENCompetence Assessment Process Specification, a task which was not part of the original work plan.

Our IMS QTI authoring tool is implemented as a JAVA Eclipse plug-in, which can be integrated into ReCourse and the planned assessment process modelling tool. It can be extended as an independent application tool as well. It is an open source software and the source code can be accessed on Sourceforge9. The tool provides a friendly user interface that enables ordinary practitioners with no knowledge of IMS QTI to define questions easily. The current version of the tool can support seven types of questions: text-based multiple-choice, text-based multiple-response, Likert, yes/no, open-question, filling-in-blank, and inline-choice. Figure 10 shows the user interface for creating an inline-choice question.

---

8 See the AQuRate website at http://aquate.kingston.ac.uk/
9 Host: tencompetence.cvs.sourceforge.net
Repository path: /cvsroot/tencompetence
Connection type: pserver
User: anonymous
Module: wp6/org.tencompetence.qtieditor
Figure 10: The user interface of IMS QTI plug-in

This example screen shot shows how the QTI plug-in offers a simple interface for authoring an inline-choice question. The interface for other types of questions follows this pattern. The results were tested in APIS and SLeD, as shown below.

Figure 11: one of a set of test items running on APIS inside SLeD
In the present version of the QTI authoring plug-in response variables are defined automatically, based on a standard template. This has the virtue of simplicity, but has the drawback that the author cannot define customized methods to process responses and produce outcomes. The definition of complex rules to calculate outcomes based on responses will always be challenging for the non-expert. In view of this we plan to develop templates to process responses which will make this practicable for ordinary users. In addition, we will support the binding between outcome variables and IMS LD properties. If a need is identified for additional interaction types, such as matching, ordering, and sliding, then these will be supported in future versions of the tool.
6 Runtime: Widget Services for Learning Design

6.1 Introduction

The solution to the provision of runtime services for IMS-LD developed by the project is centred on the use of a Widget Server, integrated into the existing IMS-LD infrastructure, as proposed in (Wilson, Sharples, & Griffiths, 2007) and outlined in D6.1. Widget type technology is currently in wide use and across most platforms in one shape or another. Examples of this technology include Apple Dashboard widgets and the gadgets found under Microsoft Vista Sidebar. For our purposes, a widget is a small software component, which can be configured to work within the context of a learning design.

Following the plans established in D6.1, a Widget Server was developed and given the name Wookie. It is responsible for providing a particular widget that the IMS Learning Design runtime system might request, so that it can be presented to the user within the Unit of Learning. As far as the user is concerned, the widget servers are completely integrated into the Learning Design Player. Despite this, the Widget Server is a separate entity and is not implemented as an integrated component of the Learning Design runtime system. This means that it can be reused in different contexts, both inside and beyond the project. Indeed we have already demonstrated Wookie widgets running inside Moodle and presenting information drawn from the TENCompetence server.

6.2 The Toolsets

The development work undertaken utilised a number of existing software toolsets, as well as the new widget service.

CopperCore

The CopperCore Learning Design Engine can import a Learning Design and orchestrate its delivery.

CopperCore Service Integration (CCSI)

CCSI is the framework which allows new services to be written, in order to extend functionality available to a Learning Design at runtime, as described in (Vogten et al., 2007).

---

10 D6.1 is available in four parts: Summary report (http://hdl.handle.net/1820/1148), Annex 1 - IMS LD Authoring, at http://hdl.handle.net/1820/1149, Annex 2: Assessment (http://hdl.handle.net/1820/1150) and Annex 3: Runtime (http://hdl.handle.net/1820/1151)

11 For information about Apple widgets see http://www.apple.com/macosx/features/dashboard/.

12 For information about Vista Sidebar see: http://www.microsoft.com/windows/products/windowsvista/features/details/sidebargadgets.mspx

13 http://coppercore.org

14 For information about CopperCore Service Integration see: http://coppercore.org & http://sourceforge.net/projects/ccsi
Service Based Learning Design Player (SLeD)\textsuperscript{15}

As described in (Weller, 2006) SLeD takes the output from the CopperCore Learning Design engine and creates a web interface. This enables a user to “play” the Learning Design, taking on a particular role or roles.

Wookie Widget Server

The Wookie Widget Server is a new standalone application developed by WP6 which supplies widgets to third parties as they are requested (such as CopperCore in this case). The server has a defined API for communication between client and server, using a combination of Javascript and AJAX\textsuperscript{16} This API is based on the W3C Widget API draft specification (W3C, 2007). The Widget Server is also extendable, in that it can import new widgets. These can then be made available to third party software.

The software components are designed to run under a J2EE container, such as JBoss. Supported platforms include Microsoft Windows, Apple MacOS and Linux.

6.3 Linking an IMS-LD service to a concrete widget service

The IMS Learning Design specification describes the following four services (as well as an additional Learning Object type)

\begin{itemize}
\item Conference
\item Monitor
\item Send Mail
\item Index Search
\end{itemize}

One of the initial design decisions to be made was to find a way to specify what particular concrete service was to be used within a specific UOL. As mentioned above, IMS-LD only includes these generic service types, which somehow need to be extended. A solution to this was formulated where existing elements and attributes of the existing IMS Learning XML binding could be used to support these specific services (such as chat for example).

Within the XML binding of each service there is the ability to allow a "parameter" value to be specified. This can be any text an author may wish, but it is usually used in the form of name-value-pairs. To use one of the widgets made available from the widget server, a parameter is added to an existing service element within an environment. The name-value pair string to enter takes the following syntax, widget=<type of widget>. So for example, to use the default chat widget service, one would enter widget=chat. Similarly to use the default forum widget service, one would enter widget=forum.

\textsuperscript{15} For information about SLeD see: http://sled.open.ac.uk/sledweb/

\textsuperscript{16} A good summary of AJAX techniques is available at: http://en.wikipedia.org/wiki/Ajax_%28programming%29
6.4 Widget Contexts and Widget Defaults

Two important concepts within the Widget Server are Contexts and Defaults. Widget Contexts simply specify the context a particular widget is to be used in. For example, one might upload a new threaded discussion forum widget into the Widget Server. In order for that widget to be used from third party software, it must be associated with a label of some sort. In the example of the threaded discussion forum widget, one might associate the label forum with that widget. One also might want to associate the label discussion with it. The Widget Server is allowing a user to associate several contexts to one particular widget. When the IMS Learning Design is authored, one can enter the widget=forum, or widget=discussion within the parameter attribute of the service, in order to associate the service with the concrete widget residing within the Widget Server. A Widget Default (or default widget) is needed because more then one widget in the system may have the same context or label. One might have two different types of chat widget within the environment. When a third party asks for a chat widget, the Widget Server must know which one to supply; hence one of the two widgets is made the default for that particular context.

6.5 Implementation of the Widget Server

The Widget Server was implemented using both the initial diagrams, and documentation from the proposed architecture(Wilson et al., 2007), along with the draft W3C Widget 1.0 specification. Initial development work started by modelling a database schema which would hold the data needed for the system. This was implemented initially in MySQL\(^\text{17}\). The business logic for handling and manipulating the data was implemented in Java\(^\text{18}\) and Hibernate \(^\text{19}\). A set of servlets\(^\text{20}\) were developed which are responsible for different parts of the Widget Server. One of the important parts of this work involved the realisation of the Widget API (W3C, 2007). The Widget API allows a widget to interact with the Widget Server, using Javascript. The widget is able to set and get values, such as preferences. More importantly, it is also has access to shared data. An example of this is a chat log between chat users, which also raises the issue that when a message is sent by one user, the other users in the chat session must also see that message. Functionality such as this created the need to include callback events. The widgets themselves are a combination of HTML and Javascript, which execute within the clients browser. Within each users browser, some of these changes need to be propagated to other users. These types of events need to call back to the application server in order to update other instances of the widget (e.g. update the chat log for other users). In order to achieve this AJAX was used, and Direct Web Remoting\(^\text{21}\) (DWR) stood out as an AJAX framework which was ideal for the task, not only in providing the callback hooks, but also in modelling the whole Widget API. DWR has some valuable AJAX

\(^\text{17}\) For information on MySQL Database see: http://www.mysql.com/
\(^\text{18}\) For information about the Java Programming language see: http://java.sun.com/
\(^\text{19}\) See Hibernate, Relational Persistence for Java and .NET: http://www.hibernate.org/
\(^\text{20}\) For information on servlets see http://java.sun.com/products/servlet/
\(^\text{21}\) For information about Direct Web Remoting see: http://getahead.org/dwr
techniques organised into an efficient library, which include the *callback* routines needed. More importantly it also allows a server side Java data model to be exposed as a set of Javascript objects, which can be manipulated within a user’s browser. The Widget API was implemented using this solution.

### 6.6 Updates to other pieces of software

The realisation of the widget system working along side CopperCore and SLeD, meant that some other changes and updates were needed in those systems. Firstly, a new service was needed within CCSI. This service is responsible for propagating the requests for a widget service from the SLeD player to the Widget Server. The CCSI service then delivers the reply to SLeD which renders the correct user interface. Additionally the SLeD player also needed to be updated, so that it could handle the `widget=context` parameter and hand that parameter off to the new CCSI service. SLeD had to be able to parse the reply from the CCSI service and show each widget it found, in a draggable, resizeable window. The solution chosen for this was to use Fenster\(^{22}\) which is a cross browser Javascript library providing exactly the functionality needed.

---

\(^{22}\) For information about Fenster see: http://www.cross-browser.com/x/lib/view.php?s=xFenster
Figure 12. Overview of the architecture

6.7 The interaction cycle of LD Runtime and Widgets

The SLeD player provides the user interface for the Unit of Learning. When a user logs in they navigate around the unit of learning until a service is encountered within a particular environment. When SLeD obtains the environment information from the CopperCore engine, it first parses each service entry found, to see if it contains any widget-context entries. If one is found, then the SLeD player builds a query. The query contains information which is specific to the run, environment, service and user who is requesting the widget; from within the unit of learning. The Widget Server uses these parameters to either return an existing widget instance (an instance which has been used before) or create a new widget instance. The CCSI widget module processes the query between the Widget Server and SLeD. Once SLeD has the response it can translate this information into the user interface. For example, returned values contain the URL of where the widget can be found and the widgets height and width to be displayed. SLeD now creates a Fenster window instance in the browser.
Ultimately, once the user clicks on the widget link in the browser, a pop up window appears containing the widget content.

Figure 13. The SLeD player with two instances of a chat widget open

6.8 Extending the widget server

Built into the Widget Server is an administrative section. Here it is possible to upload new widgets into the system by importing an archive package containing all of the widget’s resources. The archive also contains a manifest called config.xml, which must conform to the W3C widget manifest format. Once successfully imported, the administrative user can assign and create widget contexts to the widget, making the Widget Server extendable.
6.9 Widget advertising

It soon became apparent that authors had difficulty in remembering to enter the \texttt{widget-context} parameter manually within an authoring tool. Also authors had no way of knowing which widgets were available for them to use from a given Widget Server. To overcome this, a simple additional service was written into the Widget Server which \textit{advertises} which widgets it has. An authoring tool can then query the widget advert and provide the author with a list of available services. This link between Widget Service and authoring tool has been implemented within the TENCompetence IMS Learning Design authoring tool, ReCourse\textsuperscript{23}.

6.10 Service Roles

In IMS Learning Design, there exists the ability to set privileges within a conference. There are four \textit{conference roles} defined. (participant, observer, moderator, conference manager) The Widget Server can detect these role values within a Learning Design and give extra functionality to moderators and conference managers. The three bundled widgets all implement extra functionality for these users. An example of this is when a moderator uses the chat tool. The moderator can lock the widget so it cannot be updated by other users. Additionally the moderator can clear the chat log, whereas a participant cannot.

\textsuperscript{23} For information about ReCourse see: http://www.tencompetence.org/ldauthor/
6.11 Link for software

The latest version of the software can be found at http://www.tencompetence.org/ldruntime/widgetserver/ under quick start distribution. The Widget Server comes bundled with three widgets. These are chat, forum and vote tools.

References for this section


6.12 Other runtime development

6.12.1 Improvements to the interface of SLeD

In the next phase of project work it is intended to develop substitute for the SLeD Learning Design Player. Nevertheless, it has been necessary to maintain this application and adapt it so that it can play a full role in project pilots. Indeed it is this experience has indicated to the team that it is necessary to replace this component.

The presentation layer of the SLeD was updated so that the look and feel was more inline with the project colours & layout. This was instigated by first reviewing a mockup of the new SLeD, showing the new positioning and colours conceived by a designer on the project which would fit in with the other TENCompetence applications. The adaptation of SLeD involved changes to various configuration files and Stylesheets. The SLeD player incorporates a jsp header and footer file which were updated to include a new banner which showed a subtle TENCompetence logo. When the user had logged in, this header also would show the title of the Unit of Learning which was being played. The footer was simplified by removing unnecessary information.
Additionally, a main cascading style sheet (css) needed to be also updated, reflecting the project colours. Also in need of update, were some XSLT transformation files, which had to reflect the new positioning of some items. For example, there was to be a new “tabbed” resources part once a learning activity, containing multiple resources was opened. SLeD had previously handled this poorly, as it simply listed a hyperlink to a resource, which would open in a separate window. The updated version opened the link within the SLeD page and into an iframe element.

Among several other changes made in sled, a bug was fixed where all users registered with CopperCore had their usernames set to lower case at the login screen. For example, this meant that although there was a valid user in CopperCore called “Teacher” and user logged in as “Teacher”, SLeD changed this to “teacher” and this then failed on CopperCore. Another issue fixed was a problems with showing resources from a LD content package, which were nested within a subfolder. At runtime, it assumed that the resources were at the top level of the package and so some of the links to resources failed.

### 6.12.2 Enhancement of the APIS QTI engine

D6.1 provides details of the TENCompetence extension of the APIS QTI runtime engine to support QTI v.2.1. Since then work has focused on bug fixing and integrating the new APIS into the SLeD player and the CopperCore LD engine. This work involved using the CopperCore Service Integration architecture proposed in [2] and consequently updating the Adapter which enables the APIS integration into CopperCore. The integration of the QTI version 2.0 elements not originally supported by APIS has been accomplished, with support provided for the following elements required by the TENCompetence Assessment Process Specification:

a) simpleChoice  
b) multipleChoice  
c) Order  
d) Associate  
e) Match  
f) Text entry  
g) Extended text  
h) Slider

An adapter has been developed in order to support the use of QTI v2.1 tests into SLeD, without which it was not possible to calculate the overall results of tests and provide these to the CopperCore Learning Design engine.

### 6.12.3 Integration of Learning Design runtime in learn exact

The LCMS platform learn eXact Enterprise Edition is a flagship product of TENCompetence partner Giunti, for whom it was attractive to integrate IMS Learning Design because it opens up many new possibilities in eLearning, principally because it is oriented to collaborative learning. It makes it possible for Learn eXact (eLex) to

---

24 A summary work on QTI, with links to relevant documents is available at [http://www.tecn.upf.es/~daviniah/upfqti.html](http://www.tecn.upf.es/~daviniah/upfqti.html)
manage different kinds of activities, create dependencies between activities performed by different actors (such as student and teacher, but also different kinds of students). This specification is also open to social and communicative aspects. At design time it is possible to plan activities such as a chat or forum discussion. Within the eLex Platform, both the LD editor and the LD runtime will be fully developed. The editor is developed outside the TENCompetence project and currently supports level A and B of the specification and it was developed from scratch. Within the TENCompetence project, the runtime was developed starting from the existing engine CopperCore.

One of the biggest challenges was a platforms issue. eLex is completely implemented in the .NET framework and CopperCore on the contrary, is a completely JAVA application. This means that it was not possible to directly use API. For this reason the choice was to use SOAP API. This presented problems for the developers, but from the perspective of the TENCompetence project as a whole it provided an excellent case study of the feasibility of integrating the Open Source applications developed by the project with the proprietary enterprise applications run by many of the organisations which the project seeks to serve. From a high level view the integration between CopperCore and eLex is quite simple and can be described with two main blocks (CopperCore and Enterprise Lex) with a service layer between them. This approach has some performance limitation but it is the only easy way to integrate the two modules.

Another problem we encountered during the integration was the duplication of lot of information in the CopperCore repository and the LMS repository, not only relating to users, but also to the learning design packages data. The decision was to use eLex repository as the main repository and reduce at minimum the duplication. User management is handled by eLex, which means that in CopperCore users are created only when needed for run purposes. Creation, deletion, profiling etc are managed by the users module in eLex.

The approach taken for LD Packages is different. To better explain the process it is important to study the following diagram, describing the eLex overall internal architecture showing the eXact Glove component.

eLex supports content distribution and tracking according to the SCORM 1.2 and SCORM 2004 standard through the eXact Glove module it is the RTE application (Run-Time Environment). The presence of a module for the distribution of standard courses makes the platform also ideal for those scenarios where the users access the formative courses via portals (training portals, intranet or other) external from the LCMS. To benefit of this important module the platform requires that all data is stored in the platform repository.

For this reason when a new Unit of Learning is published it is pre-processed. All resources are unpacked from the zip and published one by one in the repository with their metadata. The IMS manifest is changed accordingly replacing internal referencing with URL pointing to the exact location in the portal and packed again. In this way the final LD package contains only the IMS manifest with Learning design path description, and this is the package that the platform publishes on CopperCore.
Most of the work was concentrated on the Learning Design Module, which completely new in the platform. The LD module is mainly a new web application part. It is composed of three components. The web component, the proxy module to access CopperCore service and PreProcessing Module to adapt the Learning Design Packaged to the LMS needs.

Figure 15: Learn exact and exact Glove Runtime
7 Assessment

7.1 Introduction

Many assessment needs can be addressed by the tests and questionnaires which can easily be developed using ReCourse, and which can be contextualised within Units of Learning. However, there are other more sophisticated assessment processes which need more advanced support. The Learning Design Toolkit provides the functionality which is required to author and run such processes, but authoring them is challenging even for experts, and there is no guide to the kinds of assessments which could be carried out.

To meet these needs TENCompetence has developed a formal Assessment Process Specification (ASP) for these newer forms of assessment (e.g., peer assessment and 360 degree feedback) to supplement the classic forms of assessment (e.g., multiple-choice and filling-in-blank), which are covered by IMS QTI.

In D6.1, we delivered the first version of TENCompetence Assessment Specification and the first version of the proof-of-concept Assessment Authoring Tool based on the specification. The strategy for authoring and running Units of Assessment was to use IMS LD and IMS QTI together to express the Assessment Specification. Consequently, using the proof of concept tools, we also assessed the degree to which the classic and new forms of assessment represented in the first version of the TENCompetence Assessment Specification could be transformed into an executable model represented in combination of IMS Learning Design and IMS QTI.

This experience showed that expression of the Assessment Specification as a combination of IMS LD and QTI was extremely difficult, and in some cases impossible. This was because the first version of the TENCompetence Assessment Model (see D6.1) is based on the OUNL/CITO assessment model, which is document-centric. IMS LD, on the other hand, is an activity-centric model. The mapping of the concepts and their relationships between these two models was much more difficult than was foreseen. In the period covered by this deliverable we therefore produced a second version of the TENCompetence assessment specification, now called Assessment Process Specification in order to distinguish it. This was achieved by modifying the OUNL/CITO assessment model from the perspective of process support. In addition, we developed a QTI plug-in, which is needed for editing QTI items in assessment design (as described in section 5 above).

7.2 The process oriented approach to modifying the ounl/cito model

The alternative approach that we used in modifying OUNL/CITO assessment model can be summarized as:

1) Explicitly use the concept of stage. This is used only as an underlying concept in OUNL/CITO model to organize related concepts and constitute a process model.
2) Introduce the concept of a role and reorganize related concepts such as candidate, group, and person.
3) Introduce the concept of an artefact that is produced and used in assessment including assessment test/item, response, performance, assessment results, feedback, and so on.

4) Introduce the concept of a tool to represent all applications/services used in assessment including QTI authoring tool, text editor, response tool, simulator, and so on.

5) Introduce the concept of an activity, which is treated as the central concept in the model in a way that other concepts such as role, document, tool, and stage are connected to it. Thus, the central position of assessment test/item in the model is replaced by the activity.

6) Remove some concepts related to the run-time stage (e.g., population, assessment-take, and assessment session) and some of the abstract concepts (like construction item and assessment function).

7) Introduce more assessment-specific and practitioner-familiar concepts (e.g., portfolio, concept-mapping tool, responding activity, assessor, and so on). This is treated as an open set in order to explicitly support a wide variety of forms of assessment.

The assumptions which have guided these changes are:
1) an activity-centric assessment process model can be easily integrated with IMS LD model
2) an authoring tool based on this assessment process model can be easily integrated with IMS LD authoring tool
3) an assessment process definition created by using our target assessment process authoring tool can be easily transformed into an executable model represented in IMS LD and IMS QTI.

Using this approach the second version of TENCompetence assessment specification was developed (see the following section). In order to distinguish this specification from the first version of TENCompetence assessment specification, we describe the second one the TENCompetence assessment process meta-model, and refer to it as the Assessment Process Specification (APS).

### 7.3 The Assessment Process Specification (APS)

In this deliverable we submit the conceptual model of the APS. In this section, we briefly present the conceptual model, the core of APS, through a description of its semantic aggregation model, conceptual structure model, and process structure model.

#### 7.3.1 Semantic Aggregation Model

Figure 16 represents the conceptual model of the semantic aggregation levels in APS. The model shows the levels of semantic aggregation. The semantically highest level is assessment design, which aggregates a collection of components and a method. A component can be one of five different types: role, artefact, service facility, information resource, and property. More detailed categories of each component are also depicted in Figure 16. A method consists of one or more assessment scenarios and a set of rules. An assessment scenario consists of one or more sequential stages. Each stage consists of a set of activities and/or activity-structures. Each activity-structure consists of a set of sequential, selectable, concurrent, or alternative
activities/activity-structures. A rule consists of a set of conditional expressions and a set of actions in a structuralized if-then-else/else-if format. The sub-types of each concept are illustrated in Figure 16 as well.

![Figure 16: Semantics Aggregation Model](image)

### 7.3.2 Conceptual Structure Model

Figure 17 illustrates the main structural relations between the concepts. By design, APS is an activity-centric model. The core idea is: following certain rules people with various roles perform activities/activity-structures allocated to them; they do so in stages using service facilities and information resources in order to consume and produce artefacts.
Figure 17: Conceptual Structure Model

The important attributes of an activity are the
- roles involved
- input and output artefacts
- services needed
- information resources referred to
- completion-conditions
- post-completion-actions.

For each particular type of activity, APS specifies some particular structural relations with certain types of roles, artefacts, and services listed in Figure 16. For example, a responding activity is associated with an assessee, an IMS QTI test/item, a IMS QTI player, and a response. The structural relations between these components are pre-defined in APS. Therefore, at design-time, after an activity with a certain type has been created, the associated components (e.g., roles involved, input and output artefacts, and services needed) are created automatically, together with the values of some attributes of these components (for specifying types and association relations).

7.3.3 Process Structure Model

Figure 18 illustrates the process structure relations between the seven stages (cf. Figure. 16). Usually the learning/teaching stage constitutes both the start point and end point of an integrated learning and assessment scenario. A complete process may consist of all types of stages in a sequence of learning/teaching, design, evidence collection, assessment, reflection, process, information, and learning/teaching. Sometimes one or more stages can be excluded. For example, the design stage may be excluded if the method for collecting evidence and the assessment form/criterion has been designed before the start of the execution and will be available in the execution. In a particular case, a teacher may grade students based on memory and then an evidence collection stage can be excluded. In contrast, a stage may be repeated one or more times. For example, further evidence may need to be gathered after an initial assessment; and even a design stage may be needed for creating additional assessment
items according to the user’s response at run-time. Sometimes a peer assessment can be designed in a way that enables the assesseee to review the feedback and request for elaboration. The assessor may provide further comments and detailed explanations. In some complicated cases, multiple loops may be defined within a scenario. Therefore, many concrete assessment process models can be derived from this generic process structure model.

Figure 18: Process Structure Model

The first validation study has been conducted through investigating whether the conceptual model of APS meets the requirements of completeness, flexibility, adaptability, and compatibility. The results show that the model meets these four requirements in a basis sense.

7.3.4 Representation And Transformation

Using APS, an assessment process can be represented as an assessment process model in terms of domain-specific vocabularies. Such vocabularies provide natural concepts that describe assessment processes in a way that ordinary practitioners already understand. They do not need to think of solutions in coding terms (e.g., classes, fields, and methods) or-generic concepts (e.g., activities, action objects and decision points). For representing a particular form of assessment, more restrictions and refinements are defined for practitioners to specify a particular form of assessment easily. For example, we develop a peer assessment meta-model (see Figure 19) by restricting and refining APS in a way that four types of stages are defined as a sequence and certain types of activities, artefacts, roles, and tools are associated with each stage.
Figure 19: The peer assessment meta-model

Figure 20 depicts a particular peer assessment model using the peer assessment meta-model. In this model a teacher and three learners are involved in an assessment process. The model consists of five stages. First, the teacher designs an assignment and chooses an article. Then each learner reads the article and writes an analysis report on it. When all three learners have finished their reports, they pass them on to their peers. After that, each learner reviews the transferred reports and provides feedback (including comments and grades) to the original authors. Next, each learner improves the original report on the basis of the feedback given. Finally, the tutor reviews all reports and feedback submitted by students.
After a peer assessment model has been specified using the peer assessment meta-model, it is necessary to transform this model into an executable model, represented in IMS LD and IMS QTI. The basic idea of the transformation algorithm is to create a set of instances of domain-generic concepts for each instance of a domain-specific concept and to maintain their relationships. For example, the notation for the peer assessment meta-model will be translated into a support-activity element of IMS LD with an associated environment element that will be generated together with the support-activity element. In addition, the generated environment element will contain a generated learning object element, which refers to a generated resource element, in which the assignment outcome (translated into a LD property) to be reviewed will be included and be set as visible. Moreover, an assessment form will be included in the environment in the same way.

In comparison with the IMS LD conceptual model, APS contains more assessment-specific concepts which have a high degree of specialization. As a consequence, the concepts of the APS are usually sub-classes of their corresponding concepts in IMS LD. The integration of TENCompetence assessment process model with IMS LD and IMS QTI can be illustrated as a UML class diagram. As shown in Figure 6, various colours illustrate where the concepts of APS are merged into IMS LD framework. Such an integrated model illustrates the basis of the mapping algorithm to transform an assessment process model represented in APS into an executable model represented in IMS LD and IMS QTI.

Figure 20: An exemplary peer assessment model
Figure 21: Integration of IMS LD, IMS QTI, and main concepts of APS

The exemplary peer assessment model described in Figure 21 can be transformed into a UoL\(^{25}\) which can be executed in the integrated IMS LD and IMS QTI 2.1 run-time environment developed by WP6, and any other future system which complies with these specifications.

Units of Assessment compliant with the Assessment Process Specification can be created with the proof of concept combination of ReCourse and the QTI plug-in, with Reload being used for those cases where it is necessary to use level B (for which no authoring interface is available in the current release of ReCourse). This process is too complex for practitioners without sophisticated technical knowledge. In the next phase of work we will implement an authoring tool for modeling assessment processes with APS. This tool will enable ordinary practitioners to define and customize classic and new forms of assessment as assessment process models and automatically transform the assessment process models into executable models represented in IMS LD and IMS QTI.

\(^{25}\) The code for this UoL is available at http://hdl.handle.net/1820/1292
7.3.5 Further information

The Assessment Process Specification has been described in published papers, including:

Pre-print available at http://hdl.handle.net/1820/1192

Pre-print available at http://hdl.handle.net/1820/1191

A runnable UoL of an exemplary peer assessment model, conforming to the Assessment Process Specification and expressed as a combination of IMS-LD and QTI is available for download at
http://hdl.handle.net/1820/1292